

# EUROPEAN PATENT OFFICE

## Patent Abstracts of Japan

PUBLICATION NUMBER : 2000132671  
PUBLICATION DATE : 12-05-00

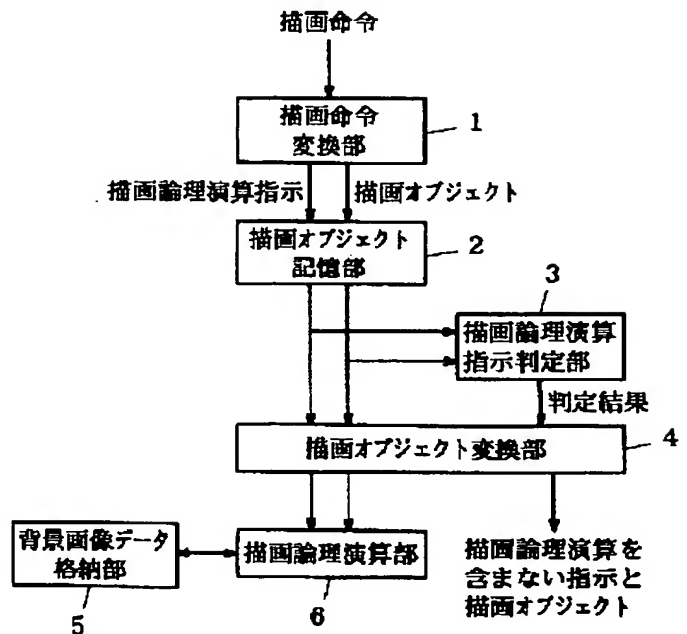
APPLICATION DATE : 04-12-98  
APPLICATION NUMBER : 10345458

APPLICANT : FUJI XEROX CO LTD;

INVENTOR : SEKINE HIROSHI;

INT.CL. : G06T 1/00 H04N 1/387

TITLE : IMAGE PROCESSOR AND IMAGE  
PROCESSING METHOD



**ABSTRACT :** PROBLEM TO BE SOLVED: To provide an image processor and an image processing method by which a plotting processing can be executed at a high speed.

**SOLUTION:** When a plotting instruction including a plotting arithmetic instruction is inputted to a plotting instruction converting part 1, the part 1 converts the inputted plotting instruction into a plotting object and the plotting arithmetic indication and stores them in a plotting object storage part 2. A plotting arithmetic instruction judging part 3 judges whether or not the plotting arithmetic instruction stored in the part 2 is provided with instruction contents to be plotted without executing a plotting arithmetic processing. When it is judged that the instruction contents can be processed without the plotting arithmetic processing as the result of judgement, a plotting object converting part 4 converts the judged plotting arithmetic instruction and the plotting object into the ones without requiring a plotting logical calculation and the processing after that is continued by another processing part.

COPYRIGHT: (C)2000,JPO

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-132671

(P2000-132671A)

(43) 公開日 平成12年5月12日 (2000.5.12)

(51) Int.Cl.<sup>7</sup>

識別記号

F I

テマコード (参考)

G 0 6 T 1/00

C 0 6 F 15/66

4 5 0

5 B 0 5 7

H 0 4 N 1/387

H 0 4 N 1/387

5 C 0 7 6

審査請求 有 請求項の数 9 O L (全 16 頁)

(21) 出願番号 特願平10-345458

(22) 出願日 平成10年12月4日 (1998.12.4)

(31) 優先権主張番号 特願平10-231759

(32) 優先日 平成10年8月18日 (1998.8.18)

(33) 優先権主張国 日本 (J P)

(71) 出願人 000005496

富士ゼロックス株式会社

東京都港区赤坂二丁目17番22号

(72) 発明者 國政 武史

神奈川県海老名市本郷2274番地 富士ゼロ

ックス株式会社内

(72) 発明者 柴田 文彦

神奈川県海老名市本郷2274番地 富士ゼロ

ックス株式会社内

(74) 代理人 100101948

弁理士 柳澤 正夫

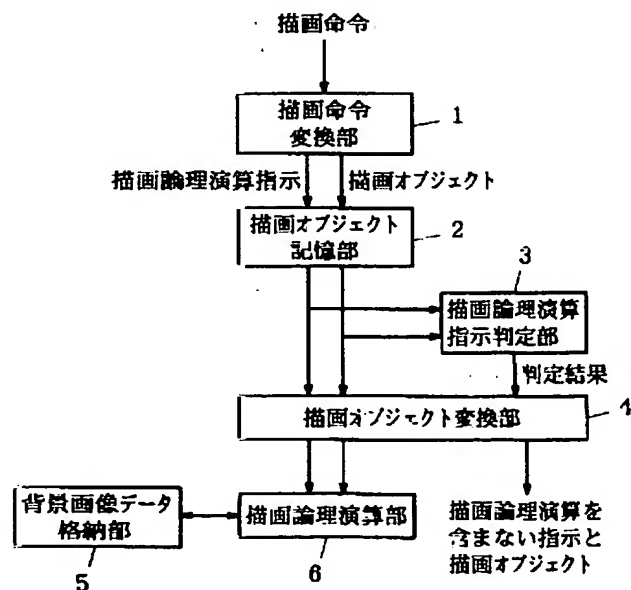
最終頁に続く

(54) 【発明の名称】 画像処理装置および画像処理方法

(57) 【要約】

【課題】 高速な描画処理を可能とした画像処理装置および画像処理方法を提供する。

【解決手段】 描画命令変換部1に描画論理演算指示を含む描画命令が入力されると、描画命令変換部1は入力された描画命令を描画オブジェクトと描画論理演算指示に変換し、描画オブジェクト記憶部2に格納する。描画論理演算指示判定部3は、描画オブジェクト記憶部2に格納されている描画論理演算指示が、描画論理演算処理を行わなくても描画できる指示内容であるかを判定する。判定の結果、描画論理演算処理を行わなくても処理可能な指示内容であると判断された場合には、描画オブジェクト変換部4によって、判定した描画論理演算指示および描画オブジェクトを、描画論理演算を必要としない指示および描画オブジェクトに変換し、他の処理部にその後の処理を継続する。



## 【特許請求の範囲】

【請求項1】 画像を構成する画素間の論理演算処理の指示を含む描画命令を受け取って処理を行う画像処理装置において、前記描画命令を描画オブジェクトおよび描画論理演算指示に変換する描画命令変換手段と、前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であるか否かを判定する描画論理演算指示判定手段と、前記描画論理演算指示判定手段において前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であると判定された場合に前記描画オブジェクトおよび描画論理演算指示を描画論理演算を必要としない指示および描画オブジェクトに変換する描画オブジェクト変換手段を有することを特徴とする画像処理装置。

【請求項2】 画像を構成する画素間の論理演算処理の指示を含む描画命令を受け取って処理を行う画像処理装置において、前記描画命令を描画オブジェクトおよび描画論理演算指示に変換する描画命令変換手段と、前記描画命令変換手段で変換された前記描画オブジェクトおよび前記論理演算指示を記憶する描画オブジェクト記憶手段と、前記描画オブジェクト記憶手段に記憶されている連続した複数の前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であるか否かを判定する描画論理演算指示判定手段と、前記描画論理演算指示判定手段において前記連続した複数の描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であると判定された場合に前記連続した複数の描画論理演算指示および前記描画オブジェクトを描画論理演算を必要としない指示および描画オブジェクトに変換する描画オブジェクト変換手段を有することを特徴とする画像処理装置。

【請求項3】 さらに、背景画像データを格納する背景画像データ格納手段と、前記背景画像データ格納手段に格納されている前記背景画像データと前記描画オブジェクトとの論理演算処理を前記描画論理演算指示に従って行い結果データを前記背景画像データ格納手段に書き戻す描画論理演算手段を有することを特徴とする請求項1または請求項2に記載の画像処理装置。

【請求項4】 前記描画オブジェクトは、形状を定義するマスクパターン、塗りつぶしパターンを定義する背景パターンのうちの少なくとも1つからなり、前記描画論理演算手段は、前記マスクパターン、前記背景パターン、および前記背景画像データのうちの指定された2つのデータに対して論理積、論理和、あるいは排他的論理和のうちのいずれかの演算を行うことを特徴とする請求項3に記載の画像処理装置。

【請求項5】 前記描画オブジェクトは、形状を定義するマスクパターン、塗りつぶしパターンを定義する背景パターンのうちの少なくとも1つからなり、前記描画論理演算手段は、前記マスクパターン、前記背景パター

ン、および前記背景画像データのうちの指定された2つのデータに対して論理積、論理和、あるいは排他的論理和の演算を組み合わせることを特徴とする請求項3に記載の画像処理装置。

【請求項6】 前記描画論理演算指示判定手段は、特定の描画論理演算指示を検出するとともに、形状を定義するマスクパターンまたは塗りつぶしパターンを定義する背景パターンが特定の条件に一致するか否かを判定することを特徴とする請求項1または請求項2に記載の画像処理装置。

【請求項7】 画像を構成する画素間の論理演算処理の指示を含む描画命令を受け取って処理を行う画像処理方法において、前記描画命令を描画オブジェクトおよび描画論理演算指示に変換し、前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であるか否かを判定し、前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であると判定された場合には、前記描画オブジェクトおよび描画論理演算指示を描画論理演算を必要としない指示および描画オブジェクトに変換することを特徴とする画像処理方法。

【請求項8】 画像を構成する画素間の論理演算処理の指示を含む描画命令を受け取って処理を行う画像処理方法において、前記描画命令を描画オブジェクトおよび描画論理演算指示に変換し、連続した複数の前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であるか否かを判定し、前記連続した複数の描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であると判定された場合に、前記連続した複数の描画論理演算指示および前記描画オブジェクトを描画論理演算を必要としない指示および描画オブジェクトに変換することを特徴とする画像処理方法。

【請求項9】 前記描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であるか否かの判定は、前記描画論理演算指示が特定の描画論理演算指示であると判定された場合に、形状を定義するマスクパターンまたは塗りつぶしパターンを定義する背景パターンが特定の条件に一致するか否かを判定するものであることを特徴とする請求項7または請求項8に記載の画像処理方法。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】本発明は、例えば画像作成装置などから出力される描画命令、特に画像を構成する画素間の論理演算処理の指示を含む描画命令を受け取って処理を行う画像処理装置および画像処理方法に関するものである。

## 【0002】

【従来の技術】パーソナルコンピュータなどの画像作成装置などから出力される様々な描画命令を処理し、印刷装置で印刷可能な画素の並びで表現された画像データに

展開する画像形成システムにおいては、描画命令を受け取ってから画像データに展開されるまでの間に描画命令に応じた様々な処理が行われる。その一つとして、複数の画像が重なる位置に描かれる場合に、もとの画像と、新たに書き込む画像との重ね合わせの方法の指示に従って、両方の画像を重ね合わせる処理がある。指示される重ね合わせの方法は、通常、論理和、論理積、排他的論理和などの論理演算の種類として指示される。このような画像間の論理演算は、一般にROP (Raster Operation) として知られている。

【0003】ROP処理は、一般に複雑で処理時間がかかり、実行のためのメモリも多く必要とするものが多い。特に、1つの効果を表現するために、複数のROP処理を組み合わせたコマンドを出力するようなアプリケーションプログラムも数多く存在する。このようなROP処理を組み合わせた指示が、印刷装置によって画像が印刷されるまでの処理時間を増大させる要因となっていた。

【0004】ROP処理の種類としては、通常、16ないし256種類が用意されている。しかしながら、これらのROP処理のすべてが常用されるわけではなく、アプリケーションプログラムによって、限られた種類のものを特徴的な組み合わせパターンによって使用する場合が多い。

【0005】ROP処理の組み合わせの例としては、3ステップのROPによって、任意形状の内側をグラデーションで塗りつぶすような処理を挙げることができる。この処理のステップ1では、任意形状を含む矩形領域の元からあった画素と、グラデーションの塗り潰しパターンとを排他的論理和 (XOR) のROP処理で演算し、目的領域の画素として描画する。次のステップ2では、任意形状を表す白黒のマスクパターンと、目的領域の画素とを論理積 (AND) のROP処理で演算し、目的領域の画素として描画する。この結果、任意形状の内側はマスクパターンの黒画素で置き換えられ、外側は白画素と論理和を取った結果、元の画素値のまま残ることになる。最後のステップ3では、ステップ1と同じ矩形領域の画素と、前述のグラデーションの塗り潰しパターンとを、再度排他的論理和 (XOR) のROP処理で演算し、目的領域の画素として描画する。この結果、ステップ2で黒く塗り潰された部分、つまり任意形状の内側部分は、当初の目的であるグラデーションパターンで塗り潰され、その外側部分は、この3ステップのROP処理を始める以前の状態に戻ることになる。

【0006】上述のような任意形状の内側を特定のパターンで塗りつぶす処理を行うためには、3回のROP処理を行わなければならない。そのため、処理に時間がかかってしまう。そこで、例えば特開平8-263674号公報にも記載されているように、このようなROP処理の特徴的パターンを検出して、ROP処理のステップ

数を削減し、より単純なROP処理に置き換えようという技術が考えられている。この文献に開示されている方法では、3ステップのROP処理のうち、最初のステップを完全に省き、ステップ2のマスクパターンと、ステップ3のグラデーションパターンとの論理積をあらかじめ演算した上で、目的領域に書き込む。これによって、ROP処理の実行回数を少なくし、高速な処理を実現している。

【0007】上述の例では3ステップのROP処理のうちの1ステップを割愛することによって高速処理を実現しているが、ROP処理を実行することには変わりがない。また、他のROP処理の組み合わせパターンについては記載されておらず、他のROP処理の組み合わせについても高速化が期待されている。

【0008】一方、上述のようなROP処理は、もともとパーソナルコンピュータ等のディスプレイ装置上の表示を対象に考え出されたものであり、論理演算の対象となる画素データは、RGB色空間で表現されていることを前提としている。一般の印刷装置では、最終的にCMYK色空間を使用しており、ROP処理が前提としている色空間とは異なっている。ROP処理は、CMYK色空間のようにK色成分が他のC、M、Y色成分に依存するような一次独立でない色空間では正しく実行することができない。そのため、CMYK色空間で表現された画素に対してROP処理を実行すると、RGB色空間の画素に対して実行した場合とは異なる結果になる。

【0009】このため、ROP処理を正しく実行するための一つの方法として、例えば、RGB色空間で表現された画像データを、はじめにRGB色空間のまま画素データに展開しながらROP処理を施す。そして、1ページ内の画像データをすべて画素データに展開処理し、ROP処理も終わった後に、1ページ分すべての画素をまとめてCMYK色空間に変換するという方法が考えられる。しかし、この方法では、RGB色空間で表現された画素データをすべて画素単位でCMYK色空間の画素に変換する必要がある。その結果、色変換に多大な処理時間を要することになる。さらに、元の描画コマンドが表す画像データの種類によらず、一律の色変換を施すことになるので、画像データの種類の適した色変換パラメータを選ぶことができず、画質上の問題も発生することになる。

【0010】このような方法に対して、いったんCMYK色空間の画素データに展開した画像データを、ROP処理が必要な部分だけCMY色空間あるいはRGB色空間に変換してからROP処理を施し、その後CMY色空間またはRGB色空間に変換した部分だけを再びCMYK色空間に変換するという方法も考えられる。この方法によれば、先の方法のようにRGB空間でいったん展開し、展開したすべての画素をそれぞれCMYK色空間に変換する場合に比べ、高速な処理が期待できる。また、

CMYK色空間のまま処理する場合と比較すれば、正しいROP処理が可能となる。しかし、CMYK色空間からCMY色空間あるいはRGB色空間への変換は1対1の変換ではなく、変換後の色値が一意に決まらないため、ROP処理のためにCMY色空間またはRGB色空間に変換した部分を再びCMYK色空間に変換しても、画素値が元に戻るとは限らない。そのため、ROP処理後にCMYK色空間に再変換したときに、ROP処理した部分とその周囲のROP処理をしなかった部分との間に色の段差ができてしまう場合がある。

【0011】このように、ROP処理を行うことによってROP処理に伴う色変換処理により色再現上の不具合が発生する場合がある。このような不具合を回避するためにも、ROP処理を少なくすることが望まれる。

【0012】

【発明が解決しようとする課題】本発明は、上述した事情に鑑みてなされたもので、高速および高画質の描画処理を可能とした画像処理装置および画像処理方法を提供することを目的とするものである。

【0013】

【課題を解決するための手段】本発明は、描画命令中の描画論理演算指示が論理演算処理を行わなくても処理可能な指示内容であるか否かを判定し、論理演算処理を行わなくても処理可能な指示内容である1個あるいは連続した複数の描画論理演算指示および描画オブジェクトを、描画論理演算を必要としない指示および描画オブジェクトに変換することを特徴とするものである。これによって、必要のない描画論理演算を行わなくて済むので、高速および高画質の描画処理が可能となる。

【0014】

【発明の実施の形態】図1は、本発明の実施の一形態を示すブロック図である。図中、1は描画命令変換部、2は描画オブジェクト記憶部、3は描画論理演算指示判定部、4は描画オブジェクト変換部、5は背景画像データ格納部、6は描画論理演算部である。描画命令変換部1は、画像を構成する画素間の論理演算処理の指示を含む描画命令を受け取って、描画オブジェクトと描画論理演算指示に変換する。描画オブジェクトは、形状を定義するマスクパターン、塗りつぶしパターンを定義する背景パターンの少なくとも1つ以上から構成される。また、論理演算指示としては、従来より知られているROP処理のいずれかが指定される。

【0015】描画オブジェクト記憶部2は、描画命令変換部1で変換した描画オブジェクトおよび描画論理演算指示を記憶する。なお、この描画オブジェクト記憶部2を配置しない構成も可能であるが、後述するように描画論理演算指示判定部3で連続する複数の描画論理演算指示について判定を行う際には、この描画オブジェクト記憶部2に複数の描画論理演算指示および対応する描画オブジェクトを記憶させておくことが便利である。

【0016】描画論理演算指示判定部3は、描画論理演算指示を解析し、その描画論理演算指示あるいは連続した複数の描画論理演算指示が、論理演算処理を行わなくても処理可能な指示内容であるか否かを判定する。この判定は、例えば描画論理演算指示が特定の描画論理演算指示である場合や、連続する複数の描画論理演算指示が特定の描画論理演算指示のパターンであることを条件とすることができる。また、この判定の際に、その描画論理演算指示によって指示されている論理演算処理の対象となる描画オブジェクトを参照して判定することもできる。

【0017】描画オブジェクト変換部4は、描画論理演算指示判定部3において論理演算処理を行わなくても処理可能な指示内容であると判定された1ないし複数の論理演算指示、その論理演算指示に対応する描画オブジェクトについて、描画論理演算を必要としない指示および描画オブジェクトに変換する。

【0018】背景画像データ格納部5は、描画論理演算部6で用いられる背景画像データを格納するとともに、描画論理演算部6で演算された結果の画像データを格納する。

【0019】描画論理演算部6は、描画オブジェクト変換部4で論理演算処理を行わなくても処理可能な指示内容であると判定された以外の描画論理演算指示について、その描画論理演算指示に従って、描画オブジェクトおよび背景画像データ格納部5に格納されている背景画像データ間の論理演算処理を実行し、その結果を背景画像データ格納部5に格納する。

【0020】上述の本発明の実施の一形態における動作の概要を説明する。描画命令変換部1に描画論理演算指示を含む描画命令が入力されると、描画命令変換部1は入力された描画命令を描画オブジェクトと描画論理演算指示に変換し、描画オブジェクト記憶部2に格納する。描画論理演算指示判定部3は、描画オブジェクト記憶部2に格納されている1つの描画論理演算指示あるいは連続する複数の描画論理演算指示が、描画論理演算処理を行わなくても描画できる指示内容であるか否かを判定する。判定の結果、描画論理演算処理を行わなくても処理可能な指示内容であると判断された場合には、描画オブジェクト変換部4によって、判定した描画論理演算指示および描画オブジェクトを、描画論理演算を必要としない指示および描画オブジェクトに変換し、他の処理部にその後の処理を継続する。

【0021】一方、描画論理演算指示判定部3によって描画論理演算処理が必要な指示内容であると判断された場合には、描画論理演算部6は、背景画像データ格納部5から背景画像データを読み出し、描画論理演算指示に従い、読み出した背景画像データと描画オブジェクトとの描画論理演算を行う。この演算が行われた結果の画像データは、背景画像データ格納部5に書き戻される。

【0022】図2は、本発明の実施の一形態を実現する画像形成システムの一例を示すブロック図である。図中、11はホストコンピュータ、12はアプリケーションプログラム、13はプリンタドライバ、14はオペレーティングシステム、15はハードウェア、16は入力装置、17は表示装置、18は外部記憶装置、19は伝送路、20はプリンタ、21はインタープリタ部、22はイメージャ部、23はレンダラ部、24はページメモリ、25はプリンタエンジン、26は制御ハードウェアである。この画像形成システムでは、ホストコンピュータ11において取得あるいは生成した画像を、プリンタ20において被記録媒体上に形成するものである。

【0023】ホストコンピュータ11は、CPUやメモリ等のハードウェア15を内蔵し、また外部機器としてキーボードやマウスなどの入力装置16、CRTなどの表示装置17、ハードディスクやCD-ROMなどの外部記憶装置18等が接続されている。さらに、インタフェースを介して伝送路19によりプリンタ20と接続されている。これらのハードウェア15や各種の装置、および、各種のソフトウェアは、オペレーティングシステム14によって制御、管理されている。このオペレーティングシステム14による制御および管理のもとで、アプリケーションプログラム12が動作する。アプリケーションプログラム12は、文字や画像などを印刷する指示を行うことができる。ここでは、印刷する内容はページ記述言語などによって記述された描画情報として出力されるものとする。

【0024】プリンタドライバ13は、プリンタ20に付属して提供されるものであり、ホストコンピュータ11にロードされて機能するプログラムである。通常、プリンタドライバ13は、特定のプリンタ20での使用が予想される様々なデータ形式にホストコンピュータ11を対応させるために使用される。プリンタドライバ13は、アプリケーションプログラム12から描画情報を受け取ると、プリンタ20に送信する前に、様々な処理ステップを実行する。例えば、アプリケーションプログラム12から受け取った描画情報をプリンタ20で解釈可能なコマンドおよび描画オブジェクトの列に変換して送信することができる。あるいは、描画情報のデータ形式のままプリンタ20へ送信してもよい。

【0025】プリンタ20では、ホストコンピュータ11から伝送路19を介して伝送されてくる描画情報あるいはコマンドと描画データの列などを受信し、被記録媒体上に画像を形成する。そのために、制御ハードウェア26、ページメモリ24、プリンタエンジン25などのハードウェアと、制御ハードウェア26上で動作するインタープリタ部21、イメージャ部22、レンダラ部23などのソフトウェアを有している。

【0026】インタープリタ部21は、プリンタ20が受信した描画情報やコマンドと描画データの列などを認

識し、コマンドと引数に組み立ててコマンドを解釈する。そして、解釈したコマンドの内容に従って、イメージャ部22を呼び出してグラフィックス描画、文字描画、イメージ処理を行う。イメージャ部22は、インタープリタ部21の指示に従って画像を描画する。レンダラ部23は、イメージャ部22で描画された画像をラスタ化し、ページメモリ24に記憶させる。このとき、例えばカラー印刷を行うのであれば、それぞれの色ごとに、例えばCMYKの4プレーンにラスタ化し、ページメモリ24に記憶させる。プリンタエンジン25は、ページメモリ24に格納された画像データに従って実際に被記録媒体上に画像を形成する。プリンタエンジン25としては、例えばレーザ方式を始め、種々の方式の記録方法を採用することができる。

【0027】このような画像形成システムにおいて、ホストコンピュータ11に導入されているプリンタドライバ13によって、図1に示した本発明の画像処理方法の全部の処理または描画オブジェクト変換部4までの処理を実行することができる。すなわち、アプリケーションプログラム12から出力される描画情報を参照し、そのうちの1つの描画論理演算指示あるいは連続した複数の描画論理演算指示が、画素間の描画論理演算を必要としない場合、その描画論理演算指示および描画オブジェクトを、描画論理演算を行わないコマンドおよび描画オブジェクトに変換してプリンタ20に送信する。この場合、描画論理演算部6および背景画像データ格納部5は、ホストコンピュータ11またはプリンタ20のいずれに設けてもよい。なお、描画オブジェクト記憶部2を設ける場合には、例えばホストコンピュータ11のハードウェア15にあるRAMや外部記憶装置18の記憶領域の一部を用いて実現することができる。また、背景画像データ格納部5をホストコンピュータ11側に設ける場合も同様である。

【0028】あるいは、プリンタ20に図1に示した本発明の画像処理装置を配置し、あるいはプリンタ20における処理として図1に示した本発明の画像処理方法を実行させることができる。この場合、プリンタドライバ13は従来と同様の処理を行えばよい。ホストコンピュータ11から送られてくる描画情報はプリンタ20における処理として、描画情報の中の1つの描画論理演算指示あるいは連続した複数の描画論理演算指示が画素間の描画論理演算を必要としない場合、その描画論理演算指示および描画オブジェクトを、描画論理演算を行わないコマンドおよび描画オブジェクトに変換すればよい。これによって、描画論理演算処理を用いずに描画オブジェクトを描画することができる。

【0029】いずれの場合にも、本発明の処理によって描画論理演算処理を減少させることができ、高速な画像形成を可能とすることができる。また、描画論理演算処理を行わないことによって、描画論理演算処理によって



生じる画質の劣化、例えば色の意図しない変化などを回避することができる。

【0030】以下、具体例を用いながら本発明の実施の一形態の動作についてさらに説明する。以下の具体例では、背景画像データ格納部5に格納されている任意のカラー画像D（以下、ディスティネーション画像と呼ぶ）と、そこに描画する描画オブジェクトであるビットマップ画像S（以下、ソース画像と呼ぶ）、そして描画オブジェクトの塗り潰しパターンを定義するパターン画像Pが假定される。このうち、ソース画像Sはマスクパターン、パターン画像Pは背景パターンとも呼ぶことができる。

【0031】ユーザは、例えばアプリケーションプログラム12のグラフィカルユーザインタフェース（GUI）を使って、これらの画像間のビットマップデータに対して、論理積（AND：a）、論理和（OR：o）、排他的論理和（XOR：x）、反転（NOT：n）等の論理演算を指定することが可能である。また、これらの論理演算を組み合わせて目的とする描画結果を得ることもできる。所望の描画結果を得るための論理演算の指定は、ユーザの直接的なオペレーションの代わりにアプリケーションプログラム12によって自動的にすることもできる。

【0032】論理演算の種類は、逆ポーランド形式で表す場合、例えば、DPaは、ディスティネーション画像Dとパターン画像Pとの論理積（a）を演算することを表している。その結果は背景画像データ格納部5に書き戻され、新たなディスティネーション画像Dとなり得る。

【0033】また、以下の第1および第2の具体例では、図1に示すような構成の本発明の画像処理方法をプリンタドライバ13に組み込んだ例について説明する。ユーザは、アプリケーションプログラム12において、例えばグラフィカルユーザインタフェース（GUI）を使用して任意に文字、グラフィックス、イメージ等の描画オブジェクトを画面上に配置してゆく。これらの描画オブジェクトは、重なりを持たせたり、透明度を指定して半透明状態を指定するなど、種々の指定を行うことができる。

【0034】これらの描画オブジェクトを印刷する場合、アプリケーションプログラム12は、オペレーティングシステム14を介してプリンタドライバ13に対して描画情報を出力する。例えば描画情報は、描画関数と色指定コード、論理演算コード、クリップ領域指定等で構成される。さらに具体的には、文字を示す描画情報は、文字であることを示す描画関数と文字の種類を示す文字コード、文字の色を示す色情報や、描画位置を示す座標情報等からなる。グラフィックスを示す描画情報は、図形の種類を示す描画関数と、論理演算コード、描画するペンの種類や、塗りつぶしのパターン指定、色情

報、描画位置を示す座標情報等からなる。また、写真等のイメージを示す描画情報は、ビットマップ情報と、論理演算コード、ビットマップの形式や、描画位置を示す座標情報等からなる。

【0035】プリンタドライバ13は、受信した描画情報から、対象とするプリンタ20に適合したコマンドおよび描画オブジェクトを生成する。このとき、描画論理演算コードを含む描画情報については、本発明の画像処理方法を適用して、描画論理演算が不要な描画論理演算コードを含む描画情報を、描画論理演算を行わないコマンドおよび描画オブジェクトに変換する。

【0036】第1の具体例では、上述の逆ポーランド形式の記法でPSDPxaxという描画論理演算を考える。図3は、描画論理演算PSDPxaxの一例の説明図である。この描画論理演算PSDPxaxは、指定されたソース画像Sが白黒イメージで表現された画像である場合、黒い部分はディスティネーション画像Dを透過し、白い部分は同時に指定されているパターン画像Pで塗りつぶす効果を狙った処理であることが知られている。図3に示す例では、矩形の画像を有するパターン画像Pと、パターン画像Pの矩形と重なりを有する位置に矩形の画像を有するディスティネーション画像Dが存在し、両画像の矩形の重なる部分に市松模様を有する2値画像であるソース画像Sが存在する。このとき、描画論理演算PSDPxaxによってソース画像Sが黒い部分はディスティネーション画像Dが、白い部分はパターン画像Pが選択され、重なり部分で市松模様となった画像D'が得られる。

【0037】しかし、この描画論理演算PSDPxaxでは、ソース画像Sが2値画像であればマスキングパターン処理によって代替可能である。図4は、マスキングパターン処理の一例の説明図である。マスキングパターン処理は、マスキングパターンMの値に応じてパターン画像Pをディスティネーション画像Dに貼り付けてゆく処理であり、論理演算処理は行わない。このマスキングパターン処理による結果は画像D'のようになる。

【0038】このマスキングパターン処理の結果である画像D'と、図3における描画論理演算処理の結果であるディスティネーション画像D'とを比較してわかるように、同じ結果が得られる。すなわち、描画論理演算PSDPxaxを指示する描画論理演算指示は、ソース画像Sが白黒2値画像であった場合には、描画論理演算を行わなくても処理可能な指示内容である。さらに、この描画論理演算指示は、マスキングパターン処理に変更可能であることがわかる。本発明では、描画論理演算指示がPSDPxaxであり、ソース画像Sが白黒2値画像である場合に、その論理演算指示およびソース画像Sをマスキングパターン処理を行う指示およびマスキングパターンMに変換して出力する。これによって、時間のかかる描画論理演算PSDPxaxを行わずに処理可能とな

る。また、この描画論理演算  $PSDP_{x \wedge x}$  を行わないことによって、画質劣化の可能性を回避することができる。

【0039】図5は、本発明の実施の一形態における第1の具体的な変換処理の一例を示すフローチャートである。S31において描画論理演算指示を含む描画情報を取得し、その中の描画論理演算コードをS32において調べる。例えば上述の描画論理演算  $PSDP_{x \wedge x}$  が描画論理演算コード184であるとすれば、描画論理演算コードが184か否かを調べる。描画論理演算コードが184であった場合には、さらにS33において、この描画情報と一緒に入力されてきたソース画像Sを解析し、ソース画像Sが白黒2値画像であるか否かを判定する。解析の結果、ソース画像Sが白黒2値画像であると判断された場合、描画オブジェクトおよび描画論理演算指示を描画論理演算を必要としない指示および描画オブジェクトに変換する。すなわちS34において、ソース画像SからマスキングパターンMを生成するとともに、パターン画像Pを使ったマスキングパターン処理を行う命令を生成し、プリンタ20に対して送信する。

【0040】一方、描画情報中の描画論理演算コードが184でなかった場合、および、描画論理演算コードが184であってもソース画像Sが白黒2値画像でなかった場合には、S35において従来と同様に処理し、例えば論理演算コード指定コマンド、パターン画像指定コマンド、ソース画像指定コマンドとしてプリンタ20に対して送信する。これらのコマンドを受信したプリンタ20側では、コマンド指示に従って、受信したソース画像、パターン画像と、ページメモリに記憶されているディステーション画像データ間の論理演算を実行してラスタデータを生成することになる。

【0041】次に第2の具体例として、別の描画論理演算について考える。ここでは、ある描画目的のために複数の描画情報を組み合わせることによって実現する場合について示す。図6は、複数の描画論理演算の組み合わせの一例の説明図である。この例では、第1の描画情報で、第1のパターン画像Pとディステーション画像Dの排他的論理和( $\wedge$ )を指定して、新たなディステーション画像D'を生成する。すなわち上述の逆ポーランド形式の記法で  $DP_{x \wedge}$  という描画論理演算である。

【0042】次に第2の描画情報で、白黒の第2のパターン画像P'と第1の描画情報によって生成したディステーション画像D'との論理積( $\wedge$ )を指定して、さらに新しいディステーション画像D''を生成する。これは  $DP_{a \wedge}$  という描画論理演算である。図6では、第2のパターン画像P'とディステーション画像D'を用いた論理演算であることを明示するため、 $D'P'a$  と示している。

【0043】最後に第3の描画情報で、第1の描画情報で指定したのと同じ第1のパターン画像Pと第2の描画

情報によって生成したディステーション画像D''との排他的論理和( $\wedge$ )を指定することによって、目的のディステーション画像D'''を生成する。この描画論理演算は、第1の描画情報の場合と同様の  $DP_{x \wedge}$  である。図6では、第1のパターン画像Pとディステーション画像D''を用いた論理演算であることを明示するため、 $D''P_{a \wedge}$  と示している。

【0044】このようにして3つの描画情報に含まれる描画論理演算指示によって行われた描画論理演算によって、第2のパターン画像P'である白黒イメージで表現された画像に対して、黒い部分はディステーション画像を透過にし、白い部分は第1と第3の描画情報で指定されているパターン画像(P)で塗りつぶした画像が得られる。

【0045】このような3段階の描画論理演算処理によって行われた処理結果は、上述の図4に示したマスキングパターン処理に他ならない。すなわち、第2の描画情報の持つ白黒2値画像のパターン画像P'をマスキングパターンMとして、黒い部分は、ディステーション画像Dを透過にし、白い部分は第1および第3の描画情報で指定した第1のパターン画像Pで塗りつぶす効果を狙った処理である。

【0046】このように、連続する3つの描画情報中の描画論理演算指示が  $DP_{x \wedge}$ 、 $DP_{a \wedge}$ 、 $DP_{x \wedge}$  の順で並び、第1の描画情報および第3の描画情報における第1のパターン画像Pが同一であり、かつ、第2のパターン画像P'が白黒2値画像である場合には、マスキングパターン処理で代替可能であることがわかる。本発明では、連続する3つの描画情報について、上述のような条件に合致するか否かを判定し、上述の条件に合致する場合にはマスキングパターン処理に変換することによって描画論理演算処理を減らすことができる。

【0047】図7は、本発明の実施の一形態における第2の具体的な変換処理の一例を示すフローチャートである。なお、ここでは描画論理演算  $DP_{x \wedge}$  を指示する論理演算コードが90、描画論理演算  $DP_{a \wedge}$  を指示する論理演算コードが160であるものとして説明する。まずS41において、第1の描画情報を取得し、S42においてその第1の描画情報中の論理演算コードが90であるか否かを判定する。論理演算コードが90であると判定された場合には、S43において、この第1の描画情報と一緒に入力されてきたパターン画像Pとその他の描画情報を例えば描画オブジェクト記憶部2などに保存しておく。これでS41で取得した描画情報に対する処理は終了する。従って、この時点においてはこの第1の描画情報に対するコマンドは出力されない。

【0048】一方、S41で取得した第1の描画情報中の論理演算コードが90でない場合には、S52において、従来と同様に、取得した第1の描画情報についてのコマンドおよび描画オブジェクトを出力して、この第1



の描画情報についての処理を終了する。

【0049】次にS44において、第2の描画情報を取得し、S45においてその第2の描画情報中の論理演算コードが160であるか否かを判定する。論理演算コードが160であった場合、さらにS46において、第2の描画情報と一緒に入力されてきた第2のパターン画像P'と座標情報を解析する。ここでの解析では、第2のパターン画像P'が白黒の2値画像であるか否かと、前に保持した描画情報との対象とする座標情報を判断する。解析の結果、第2のパターン画像P'が白黒2値画像であり、なおかつ対象とする座標が同一であると判断された場合、S47において第2のパターン画像P'とその他の描画情報を描画オブジェクト記憶部2などに保存しておく。この場合も、この第2の描画情報に対するコマンドを出力せずに、第2の描画情報に関する処理を終了する。ここまでの処理で、2つの描画情報が描画オブジェクト記憶部2に保持されている。

【0050】一方、S44で取得した第2の描画情報中の論理演算コードが160でなかった場合と、S46において第2のパターン画像P'が白黒2値画像ではないと判断された場合、あるいは、対象とする座標が異なった場合には、S53において、描画オブジェクト記憶部2に保存された第1の描画情報からコマンドおよび描画オブジェクトを生成してプリンタ20に送信する。次にS54において、第2の描画情報に基づいてコマンドおよび描画オブジェクトを生成してプリンタ20に送信する。この場合のコマンドは、通常の描画コマンドとしてプリンタ20で処理される。

【0051】次にS48において、第3の描画情報を取得して、S49においてその第3の描画情報中の論理演算コードが90であるか否かを判定する。論理演算コードが90であった場合、さらにS50において、第3の描画情報と一緒に入力されてきた第3のパターン画像P''と座標情報を解析する。この解析では、この第3のパターン画像P''が、第1の描画情報に対応する第1のパターン画像Pと同一なものであるか否かと、対象とする座標が保持している第1の描画情報の座標と一致するかを判断する。解析の結果、第3のパターン画像P''が、第1のパターン画像Pと同一であり、なおかつ対象とする座標も一致すると判断された場合、ここまでの一連の第1ないし第3の描画情報は、マスキングパターン処理に変換可能であると判断する。そしてS51において、第2の描画情報に対応する第2のパターン画像P'をマスキングパターンMに変換するとともに、指定するパターン画像として第1または第3の描画情報に対応する第1のパターン画像Pまたは第3のパターン画像P''としたマスキングパターン処理を行う命令としてコマンドを生成し、プリンタ20に対して送信する。この場合も、プリンタ20における描画はマスキングパターンに従って、パターンイメージをディステーション画像

に対して貼り付けてゆく処理であり、各画像データ間での論理演算処理は必要なくなる。

【0052】一方、S48で取得した第3の描画情報中の論理演算コードが90でなかった場合と、S50において、第3のパターン画像P''が第1のパターン画像Pと一致していなかった場合、あるいは対象とする座標情報が異なっていた場合には、第1ないし第3の描画情報それぞれについてコマンドおよび描画オブジェクトを生成して出力する。すなわち、S55において、描画オブジェクト記憶部2に保存されている第1の描画情報からコマンドおよび描画オブジェクトを生成してプリンタ20に送信する。次にS56において、同じく描画オブジェクト記憶部2に保存されている第2の描画情報に基づいてコマンドおよび描画オブジェクトを生成してプリンタ20に送信する。最後にS57において、第3の描画情報に基づいてコマンドおよび描画オブジェクトを生成してプリンタ20に送信する。この場合の各コマンドは、通常の描画コマンドとしてプリンタ20で処理されることになる。

【0053】上述の第1および第2の具体例では、それぞれ、描画論理演算PSDPxax、あるいは、連続する3つの描画論理演算DPx、DPa、DPxを抽出した。これらは一例であって、他の論理演算処理を行わなくても処理可能な1ないし連続した複数の描画論理演算指示について、同様に論理演算処理を含まない指示および描画オブジェクトに変換することができる。

【0054】また、上述のような第1および第2の具体例についての処理は単独で組み込むほか、組み合わせで組み込んでもよい。もちろん、他の論理演算処理を行わなくても処理可能な1ないし連続した複数の描画論理演算指示とともに判定し、処理してもよい。例えば、ある描画情報を取得したとき、その描画情報中の論理演算コードが184であるのか、90であるのか、あるいはその他の変換対象の論理演算コードであるのかを判定し、論理演算コードが184であれば図5に示す処理を、論理演算コードが90であれば図7に示す処理を行う等といったように、論理演算コードに応じた処理を行うように構成することが可能である。

【0055】また上述の第1および第2の具体例では、オペレーティングシステム14を介したアプリケーションプログラム12からの描画情報を、特定の論理演算コードを持った描画情報についてプリンタドライバ13の内部で解析を行い、画像データ間の論理演算を必要としないコマンドを生成した。しかしながら、このような解析とコマンドの変換処理は、プリンタドライバ13の内部での処理に限定するものではない。すなわち、プリンタドライバ13の内部では通常の描画コマンドとして処理を行い、その結果として、特定の論理演算を指定するコマンドもプリンタ20へ送信する。そしてプリンタ20側において、上述のような解析を例えばインタープリ

タ部21において実行することによって、結果として、論理演算を必要としない内部処理に変換することも可能である。この場合、図1に示した描画命令は、プリンタ20が受け取るコマンド、あるいはインタプリタ部21で生成した内部コードとすることができる。

【0056】以下、第3の具体例について説明する。この第3の具体例では、図1に示すような構成の本発明の画像処理方法をプリンタ20の内部で、インタプリタ部21に組み込んだ例について説明する。ユーザは、アプリケーションプログラム12において、例えばグラフィカルユーザインタフェース(GUI)を使用して任意に文字、グラフィックス、イメージ等の描画オブジェクトを画面上に配置していく。これらの描画オブジェクトは、重なりを持たせたり、半透明状態を指定するなど、種々の指定を行うことができる。

【0057】これらの描画方法のなかで、とくに、複数のグラフィックスデータを用いてグラデーション画像を作成することができる。図8は、論理演算を使用したグラデーション画像の生成方法の一例の説明図である。図8(A)は、グラデーションを描画する前のディスティネーション画像Dを示しており、三角形の図形が描かれている例を表している。次に、図8(B)では、ソース画像S1からソース画像S5まで、SDxの演算、すなわちディスティネーション画像Dとソース画像Sの排他的論理和を行って描画する。次に、図8(C)では、値0の色(RGB色空間では黒)によって最終的にグラデーション部分を残したい形状で描画する。この例では、ソース画像S6として示す楕円形状の図形を描画している。最後に、図8(D)では、図8(B)と同様な描画を行う。すなわち、ソース画像S1からソース画像S5を使用して、SDxの演算を行って描画する。

【0058】ここで、排他的論理和は、同一の位置に同一の色で2回描画すると、処理を行う前の画素を保存する性質がある。また、値が0である画像に対しての排他的論理和は、単純に上書き描画した場合と同様な結果となる。図9は、排他的論理和の演算結果の説明図である。図9(A)は、ディスティネーション画像Dに対し、同一のソース画像Sで2回排他的論理和を行った場合を示している。すなわち、ディスティネーション画像Dとソース画像の排他的論理和はDSxであり、その結果に対してソース画像Sで排他的論理和を演算することはSDSxxで示される。図9(A)では、ディスティネーション画像Dとソース画像Sのすべての取りうる値の組み合わせを示している。図9(A)からわかるように、SDSxxの演算結果は、もとの画像であるディスティネーション画像Dと一致している。

【0059】また、図9(B)には、値が0のディスティネーション画像Dに対し、ソース画像Sと排他的論理和をとった場合を示している。このときの演算はDSxである。図9(B)に示すように、DSxの演算結果と

ソース画像Sは一致する。

【0060】この結果、図8(C)に示す画像に対してソース画像S1からソース画像S5までの画像で排他的論理和を取ることににより、値0の色によって描画した楕円形状の部分以外の部分では、図9(A)で説明したようにもとのディスティネーション画像Dが現れる。また、楕円形状の部分では図9(B)で説明したようにソース画像S1からソース画像S5までの画像が現出する。すなわち、図8(D)では、ちょうど図8(C)の処理で値が0の色で書き込んだ楕円形状の部分だけグラデーションがクリップされた結果として出力されることになる。このような描画方法によれば、どのような形状であっても自由にグラデーションを描画することが可能となる。

【0061】この一連の処理を表す描画命令をプリンタドライバ13がアプリケーションプログラム12から受け取って、通信路19を介してプリンタ20内部のインタプリタ部21に送る場合、一般的には、PDL(ページ記述言語)と呼ばれる描画命令群により送られる。図10は、ページ記述言語の一例の説明図、図11は、ページ記述言語において用いられる論理演算コードの一例の説明図である。ここでは、setattribute命令、drawrect命令、drawellipse命令、startclip命令、endclip命令について示している。

【0062】setattribute命令は、以降に描画される図形の属性を設定する。setattribute命令のパラメータ数は4であり、1から3番目のパラメータは、RGB色空間での色値をR、G、Bの順番で表す。4番目のパラメータは、論理演算のコードを表している。論理演算と論理演算のコードとの対応は、図11に示す通りであり、1から16までの16種類が定義されている。

【0063】また、drawrect命令は、矩形形状の図形を、現在の属性に従って塗りつぶして描画する。drawrect命令のパラメータ数は4であり、2つの対角点のx、及びy座標を表す。また、drawellipse命令は、楕円形状の図形を、現在の属性に従って塗りつぶして描画する。drawellipse命令のパラメータ数は4であり、楕円形状を取り囲む外接矩形の2つの対角点のx、及びy座標を表す。

【0064】また、startclip命令、endclip命令は、クリップ領域を作成するための命令であり、パラメータを必要としない。この2つの命令で囲まれたステートメント内部で発行された図形描画命令は、クリップ領域へと変換される。クリップ処理とは、ある図形を任意の形に切り取る処理であり、例えば特開平6-274643号公報に示される手法などによって実現することができる。

【0065】図12は、第3の具体例を描画するための

描画命令群の一例の説明図である。図12(A)は、図10、図11で説明したように定義されるPDLを用いて、図8に示したグラデーション画像の描画を記述した場合を示している。10行目までのsetattribute命令とdrawrect命令は、図8(B)の描画に対応している。すなわち、2つの命令ごとにソース画像S1からソース画像S5を描画している。次の11及び12行目に記述されている命令は、RGB空間での色をR、G、B=0、0、0とした楕円形状を描画する命令である。この描画は、図8(C)に示した楕円形状の描画に対応している。さらに13行目から最終行までの命令は、ソース画像S1からソース画像S5を描画する命令であり、図8(D)における描画に対応する。

【0066】これらの命令群は、図8(C)において描画した図形形状(この例では楕円形状)が、最終的に残したいグラデーション図形の形状となる。すなわち、グラデーション図形を、図8(C)において描画した図形形状でクリップ処理した場合と等価である。これを利用し、startclip命令、endclipを用いると、図12(A)に示したPDLは、図12(B)のように書き換えることが可能となる。図12(B)に示す記述では、はじめに1〜3行目において、図8(C)における楕円形状の描画部分をクリップ領域として設定する。その後に、4〜13行目において、一連のグラデーション図形を描画する。この4〜13行目の記述は、図12(A)における1〜10行目あるいは13〜22行目と同じであるが、論理演算コードを13番(ソース画像による上書き)に変更している。

【0067】このように、図12(A)では、論理演算として排他的論理和を使用して図8に示したように描画したが、図12(B)ではクリップ命令を用いることによって、全て、通常の上書き描画で表現できることがわかる。本発明では、値が0の図形の描画を挟んで同じソース画像を排他的論理和により描画する命令が並んでいる場合、クリップ命令を用いた描画命令列に変換することによって論理演算を行わない描画命令列に変換する。これによって描画処理を高速化することが可能になる。また、論理演算を行わないことによって、画質の劣化を回避することができる。

【0068】図13ないし図15は、本発明の実施の一形態における第3の具体的な変換処理の一例を示すフローチャートである。図13および図14で示す処理は、内部状態としてaからdの状態遷移を行う。状態aは初期状態を表す。状態bは、例えば図8(B)に示すソース画像S1からソース画像S5に対応する描画命令を受け取っている状態、つまり、図12(A)の10行目までの命令とパラメータのセットを受け取る状態を表す。また、この状態bで受け取る命令とパラメータのセットの数をCOUNT1変数に記憶する。状態cは、図8(C)に示す値が0の色の楕円形状を描画する描画命令

を受け取っている状態、つまり、図12(A)の11、12行目の命令とパラメータのセットを受け取っている状態を表す。また、この状態cで受け取る命令とパラメータのセットをCOUNT2変数に記憶する。状態dは、図8(D)に示すソース画像S1からソース画像S5に対応する描画命令を受け取っている状態、つまり、図12(A)の13行目以降を受け取っている状態を表す。また、この状態dで受け取る命令とパラメータのセットをCOUNT3変数に記憶する。さらに、これらCOUNT1からCOUNT3の各変数に対応した命令とパラメータは、それぞれバッファ1からバッファ3へと格納されるものとする。

【0069】まず処理開始直後のS61において、内部状態をaに初期化するとともに、COUNT1からCOUNT3までの各変数を0に初期化する。次のS62において、PDLの読み込みを継続するか否かを判定する。これは、PDLが印刷命令を発行していたり、何らかの理由で処理継続を中断する場合の判定を行う。読み込みを継続する場合には、S63において、コマンドおよびパラメータのセットを1つ読み込む。

【0070】S64において、読み込んだコマンドおよびパラメータがsetattribute命令であるかを判定する。setattribute命令である場合には、S65において、4番目のパラメータである論理演算コードが7番、すなわちDSx演算を指定しているか否かを判定する。DSx演算を指定している場合には、さらにS66において、状態がaであるか否かを判定し、状態がaであれば新たにDSx演算の命令列が始まったものとして、S67において状態をbに変更する。またS68において、状態がcであるか否かを判定し、状態がcであれば値が0の色の図形を描画した後のDSx演算の命令列が始まったものとして、S69において状態をdに変更する。状態がそれ以外の場合には、状態の変更を行わずにS75へ進む。

【0071】setattribute命令の4番目のパラメータが7番ではない場合にはS70に進み、同じく4番目のパラメータが13番、すなわちソース画像Sとして上書きを指定しているか否かを判定する。上書きを指定している場合、S71において状態がbで、色の値が0か否かを判定する。この場合、DSx演算の描画命令列の後にクリップすべき図形を描画している可能性がある。そのため、S72において状態をcに変更してS75に進む。また、S73において状態がcで、色の値が0の場合には、クリップすべき図形の描画が続いているものとして、状態を変更せずにS75へ進む。これ以外の状態、あるいは色の値が0以外の場合には、クリップ命令への変換を行うことができないので、S87へ進んでそれまでのPDLをそのまま出力する。また、setattribute命令の4番目のパラメータが7番でも13番でもない論理演算が指定されている場合

も、クリップ命令への変換の対象外であるので、S87へ進んでそれまでのPDLをそのまま出力する。

【0072】S63で読み込んだ命令がset attribute命令ではない場合、S74において、図形形状の描画命令であるかを判定する。drawrect命令やdrawellipse命令などのような図形形状の描画命令である場合には、命令をバッファにそのまま格納すべく、S75へ進む。set attribute命令でも、図形形状の描画命令でもない命令が現れた場合には、クリップ命令への変換を行わず、S87へ進んでそれまでのPDLをそのまま出力する。

【0073】S63で読み込んだ命令がset attribute命令であり、DSx演算(7番)を指定している場合、あるいは、上書き(13番)を指定するとともに色の値が0でかつ状態がb、cの場合と、図形形状の描画命令である場合には、S75～S83において、命令をバッファに格納する。すなわち、S75において状態がbであると判定される場合には、S76においてCOUNT1変数を1増加させ、S77において命令をバッファ1に格納する。同様に、S78において状態がcであると判定される場合には、S79においてCOUNT2変数を1増加させ、S80において命令をバッファ2に格納する。S77、S80において命令をバッファ1またはバッファ2に格納した後、S62へ戻り、読み込みの継続を判定した後、次の命令についての処理を進める。

【0074】さらに、S81において状態がdであると判定される場合には、S82においてCOUNT3変数を1増加させ、S83において命令をバッファ3に格納する。この場合には、S84においてCOUNT3変数の値とCOUNT1変数の値を比較し、COUNT3変数の値がCOUNT1変数の値よりも小さい、すなわち、値が0の色の図形の描画の前に存在していたDSx演算の命令数よりも少ない場合には、S62へ戻って次の命令の処理を行う。COUNT3変数の値がCOUNT1変数の値が等しくなると、これをS85で検出し、クリップ命令による置き換えが可能な命令列を検出したものとして、S86において実際に置き換えが可能かを判定し、可能であれば置き換え処理を行う。S86の処理後は、新たな置き換え可能な命令列を検出すべく、S61へ戻る。

【0075】なお、COUNT3変数の値がCOUNT1変数の値よりも大きくなってしまったときには、クリップ命令への変換を行うことができないので、S87へ進んでそれまでのPDLをそのまま出力する。また、状態がaの場合にも、バッファに格納することなく、S87においてそれまでのPDLをそのまま出力する。S87においてPDLを出力した後は、S61へ戻り、新たな置き換え可能な命令列を検出すべく処理を繰り返す。

【0076】最後に、読み込むべき描画命令が無くなる

など、読み込みを継続しない場合には、S88においてそれまでのPDLをそのまま出力し、処理を終了する。

【0077】S86において行う、実際に置き換えが可能かを判定し、可能であれば置き換え処理を行う処理を図15に示している。S91において、バッファ1とバッファ3の内容を比較する。この場合、バッファ1およびバッファ3に格納されている命令の数はCOUNT1変数あるいはCOUNT3変数により示されている。S92において、S91での比較の結果、バッファ1の内容とバッファ3の内容が一致しているかを判定する。一致している場合には、同じソース画像を排他的論理和で2回論理演算を行っていることを示している。また、図13、図14の処理において、これらの排他的論理和の命令列の間に、値が0の色による図形の描画が行われていることもわかっている。そのため、検出した命令列はクリップ命令を用いて書き換えが可能であると判断することができる。

【0078】この判断を受け、S93において、クリップ命令を用いた描画命令列に変換する。すなわち、startclip命令を出力し、バッファ2に格納されている図形形状の描画命令(コマンド/パラメータ)を出力し、endclip命令を出力する。さらに、バッファ1に格納されているすべての命令(コマンド/パラメータ)に対して論理演算コードを13番(ソース画像として上書きする演算)に変更して出力する。このようにして、論理演算を用いない描画命令列に変換することができる。

【0079】なお、S92において、バッファ1の内容とバッファ2の内容が一致しない場合には、同じソース画像を2回排他的論理和により演算しているとは言えないので、変換処理を行わずに、S94において蓄積されているすべてのPDLをそのまま出力する。

【0080】上述の変換処理を、図8で示した具体例を描画する描画命令列、すなわち図12(A)に示すPDLを図12(B)に示すPDLに変換する場合について説明する。図12(A)では、まず1行目の「set attribute 30 30 30 7」を読み込むことになる。この場合、命令はset attribute命令であり、4番目のパラメータはDSx演算(7番)である。また、初期状態はaであるので、S67において、状態をbに変更する。状態がbに変更されているので、S76においてCOUNT1変数に1を加え、S77においてこの1行目の命令をバッファ1に格納する。そしてS62へ戻る。

【0081】次の命令、つまり図12(A)における2行目の「drawrect 50 10 70 90」を受け取る。この場合、set attribute命令ではないが、図形形状の描画命令である。また、状態はbであるので、S76においてCOUNT1変数に1を加え、S77において2行目の命令をバッファ1に格納す

る。そしてS62へ戻る。

【0082】次の3行目の命令は、setattribute命令であり、4番目のパラメータはDSx演算（7番）であるが、状態はbであるので状態の変更を行わず、S76においてCOUNT1変数に1を加え、S77においてこの3行目の命令をバッファ1に格納する。そしてS62へ戻る。

【0083】このようにして、図12（A）の10行目までは、この2つの流れを交互に繰り返すことになる。これにより、バッファ1に10行目までの描画命令が格納される。

【0084】次に、11行目の「setattribute 0 0 0 13」を読み込むと、setattribute命令ではあるが、4番目のパラメータはDSx演算（7番）ではない。しかし、4番目のパラメータはソース画像の上書きを示す13番の論理演算コードであるので、さらに状態と色を判定する。この命令の時点では状態はbであり、この命令の描画色の値はR、G、B=0、0、0である。そのため、S72において状態をcに変更する。状態がcに変更されているので、S79においてCOUNT2変数に1を加え、S80においてこの11行目の命令をバッファ2に格納する。そしてS62へ戻る。次の12行目のdrawellipse命令は、図形形状を描画する命令であるので、S79においてCOUNT2変数に1を加え、S80においてバッファ2に格納される。

【0085】次に、13行目の「setattribute 30 30 30 7」を読み込むと、命令がsetattribute命令であり、4番目のパラメータはDSx演算（7番）であり、さらに状態がcであるので、S69において状態をdに変更する。状態がdに変更されているので、S82においてCOUNT3変数に1を加え、S83においてこの13行目の命令をバッファ3に格納する。そしてS62へ戻る。以下、14行目から22行目の命令についても、S82においてCOUNT3変数に1を加え、S83においてバッファ3に格納される。

【0086】22行目の命令までを処理した時点で、COUNT1=10、COUNT2=2、COUNT3=10となっている。COUNT1変数の値とCOUNT3変数の値が等しいので、図15に示す処理を実行する。

【0087】図12（A）に示す描画命令列では、1～10行目の命令列と、13～22行目の命令列は等しいので、クリップ処理を用いた命令列に変換できると判断される。そして、S93において、次の処理を行う。

（A）startclip命令を出力する。

（B）バッファ2の図形描画の命令とパラメータを全て出力する。

（C）endclip命令を出力する。

（D）バッファ1の命令とパラメータを全て出力する。この時、setattribute命令のパラメータで論理演算を指定する4番目のパラメータには13番（ソース画像の上書き描画）に変更して出力する。

以上の（A）から（D）の4つの処理を実行することで、図12（A）に示される論理演算を使用した一連のグラデーション描画命令は、図12（B）に示すように置換される。このようにして、論理演算を含む図12（A）に示す描画命令列を、論理演算を含まない図12（B）に示す描画命令列に変換することができる。これにより、描画処理速度を向上させることができる。

【0088】なお、この第3の具体例では、図1におけるプリンタ20の内部で変換処理を行うものとして説明した。しかしこれに限らず、上述の第1、第2の具体例と同様に、プリンタドライバ13で上述のような変換処理を行ってもよい。また、この第3の具体例も、上述の第1および第2の具体例とともに、組み合わせて実行できるように構成することも可能である。

【0089】また、上述の第1ないし第3の具体例では、特に図2に示したような画像形成システムをもとに説明したが、本発明はこれに限らず、画像を形成する装置を含む任意のシステム構成において適用可能である。

【0090】

【発明の効果】以上の説明から明かなように、本発明によれば、論理演算処理を含む描画命令について、指示された論理演算処理をそのまま実行することが必要でない場合に、論理演算処理を含まないコマンドに変換して実行することができるので、論理演算処理にかかる時間を省いた高速な画像処理が可能となる。さらに、論理演算処理を行わないことによって、論理演算処理に伴う色変換処理の結果生じる画質上の不具合を回避することができるので、結果として高速な処理と高画質を両立した画像処理が実現できるという効果がある。

【図面の簡単な説明】

【図1】 本発明の実施の一形態を示すブロック図である。

【図2】 本発明の実施の一形態を実現する画像形成システムの一例を示すブロック図である。

【図3】 描画論理演算PSDPmaxの一例の説明図である。

【図4】 マスキングパターン処理の一例の説明図である。

【図5】 本発明の実施の一形態における第1の具体的な変換処理の一例を示すフローチャートである。

【図6】 複数の描画論理演算の組み合わせの一例の説明図である。

【図7】 本発明の実施の一形態における第2の具体的な変換処理の一例を示すフローチャートである。

【図8】 論理演算を使用したグラデーション画像の生成方法の一例の説明図である。

【図9】 排他的論理和の演算結果の説明図である。

【図10】 ページ記述言語の一例の説明図である。

【図11】 ページ記述言語において用いられる論理演算コードの一例の説明図である。

【図12】 第3の具体例を描画するための描画命令群の一例の説明図である。

【図13】 本発明の実施の一形態における第3の具体的な変換処理の一例を示すフローチャートである。

【図14】 本発明の実施の一形態における第3の具体的な変換処理の一例を示すフローチャート（続き）である。

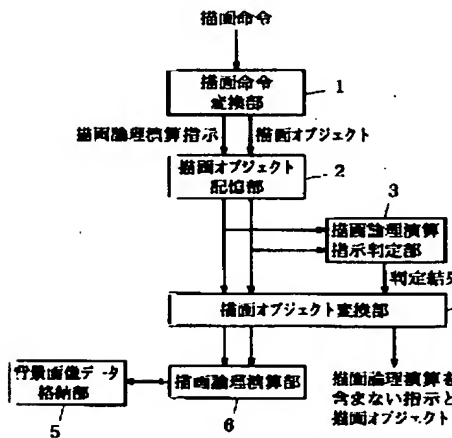
【図15】 本発明の実施の一形態における第3の具体的な変換処理の一例を示すフローチャート（サブルーチン）である。

ン）である。

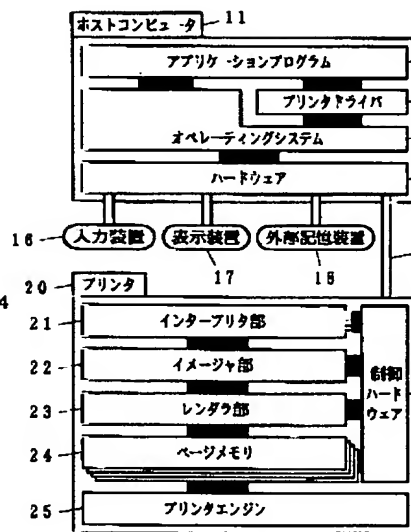
【符号の説明】

1…描画命令変換部、2…描画オブジェクト記憶部、3…描画論理演算指示判定部、4…描画オブジェクト変換部、5…背景画像データ格納部、6…描画論理演算部、11…ホストコンピュータ、12…アプリケーションプログラム、13…プリンタドライバ、14…オペレーティングシステム、15…ハードウェア、16…入力装置、17…表示装置、18…外部記憶装置、19…伝送路、20…プリンタ、21…インタープリタ部、22…イメージ部、23…レンダラ部、24…ページメモリ、25…プリンタエンジン、26…制御ハードウェア。

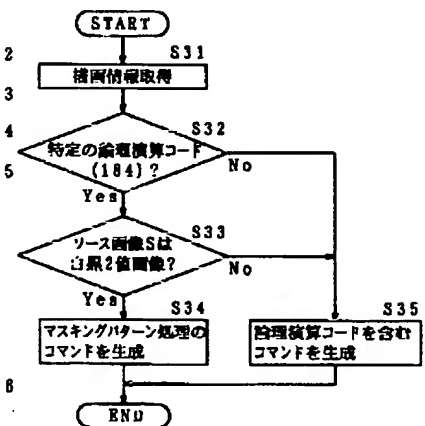
【図1】



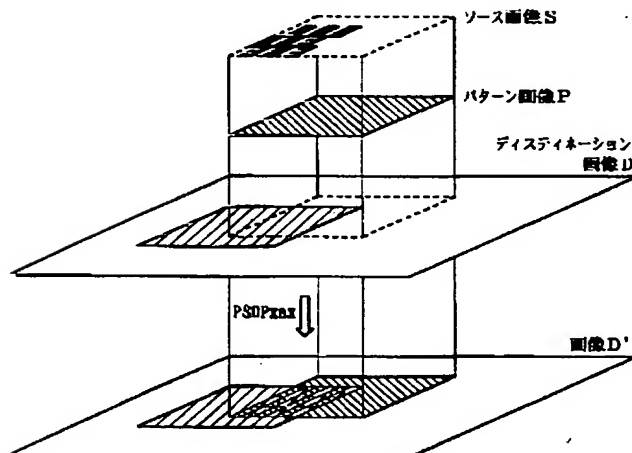
【図2】



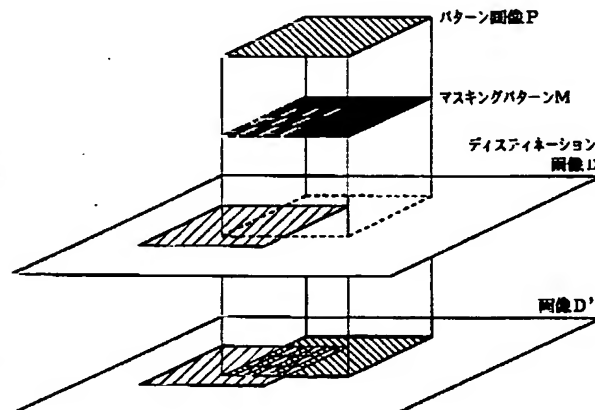
【図5】



【図3】

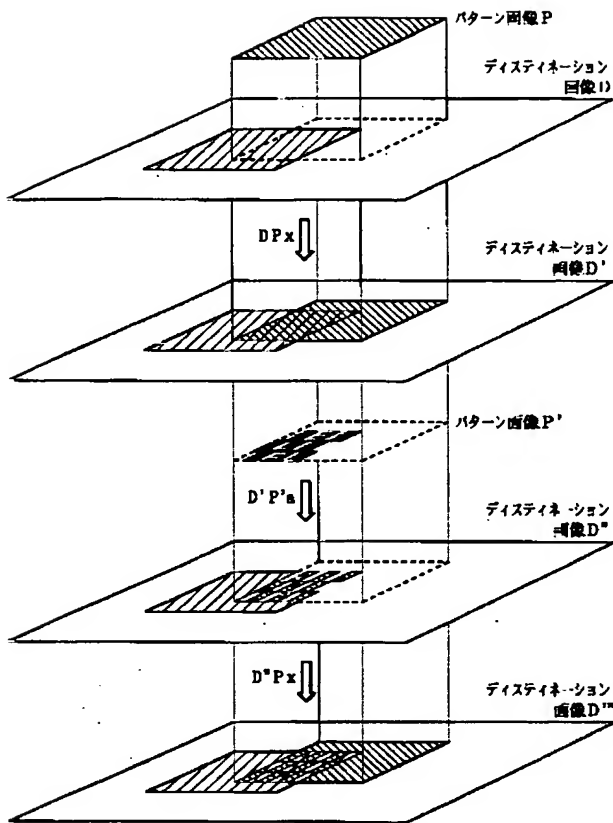


【図4】

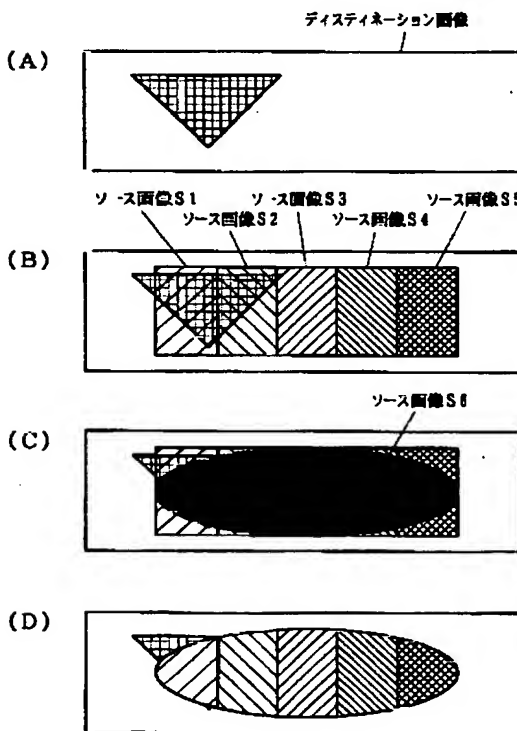




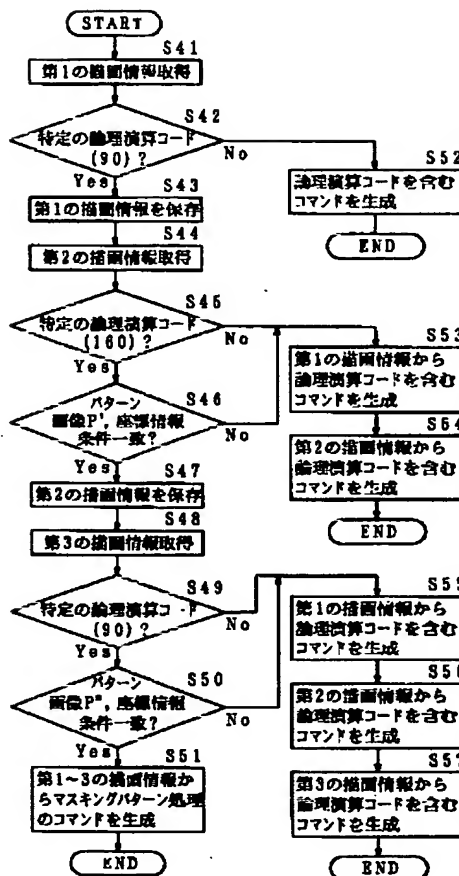
【図6】



【図8】



【図7】



【図11】

論理演算	コード
0	1
DSon	2
DSna	3
DSn	4
DSna	5
Dn	6
DSx	7
DSan	8
DSa	9
DSxn	10
D	11
DSno	12
S	13
SDno	14
DSO	15
1	16

【図9】

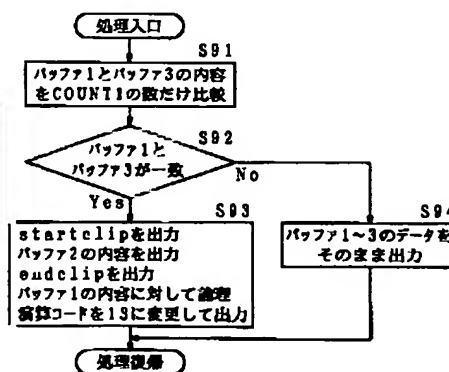
**(A)**

D	1	1	0	0
S	1	0	1	0
DS <sub>x</sub>	0	1	1	0
SDS <sub>xx</sub>	1	1	0	0

**(B)**

D	0	0
S	1	0
DS <sub>x</sub>	1	0

【図15】



【図10】

コマンド名	パラメータ
setattribute	c1 RGB色空間のRの値 c2 RGB色空間のGの値 c3 RGB色空間のBの値 code 論理演算コード
drawrect	x1 対角点1のx座標 y1 対角点1のy座標 x2 対角点2のx座標 y2 対角点2のy座標
drawellipse	x1 外接矩形の対角点1のx座標 y1 外接矩形の対角点1のy座標 x2 外接矩形の対角点2のx座標 y2 外接矩形の対角点2のy座標
startclip	なし
endclip	なし

【図12】

(A)

```

1 setattribute 30 30 30 7
2 drawrect 50 10 70 90
3 setattribute 60 60 60 7
4 drawrect 70 10 90 90
5 setattribute 80 90 90 7
6 drawrect 90 10 110 90
7 setattribute 120 120 120 7
8 drawrect 110 10 130 90
9 setattribute 160 160 160 7
10 drawrect 130 10 150 90
11 setattribute 0 0 0 13
12 drawellipse 60 20 140 80
13 setattribute 30 30 30 7
14 drawrect 50 10 70 90
15 setattribute 60 60 60 7
16 drawrect 70 10 90 90
17 setattribute 80 90 90 7
18 drawrect 90 10 110 90
19 setattribute 120 120 120 7
20 drawrect 110 10 130 90
21 setattribute 160 160 160 7
22 drawrect 130 10 150 90

```

グラデーション図形  
排他的論理和演算

値が0の色の図形描画  
(楕円形状を上書き)

グラデーション図形  
排他的論理和演算

(B)

```

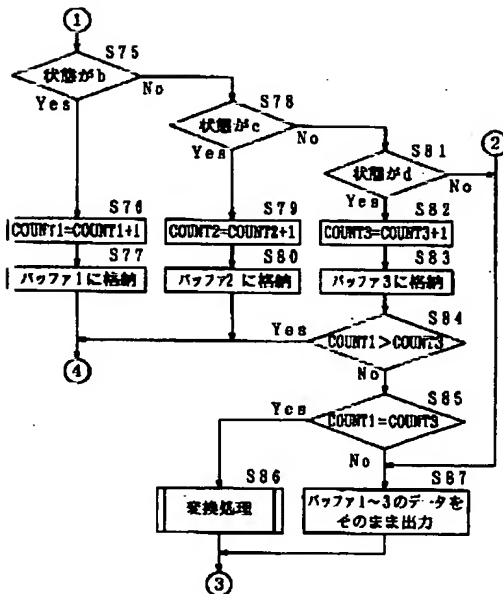
1 startclip
2 drawellipse 60 20 140 80
3 endclip
4 setattribute 30 30 30 13
5 drawrect 50 10 70 90
6 setattribute 60 60 60 13
7 drawrect 70 10 90 90
8 setattribute 80 90 90 13
9 drawrect 90 10 110 90
10 setattribute 120 120 120 13
11 drawrect 110 10 130 90
12 setattribute 160 160 160 13
13 drawrect 130 10 150 90

```

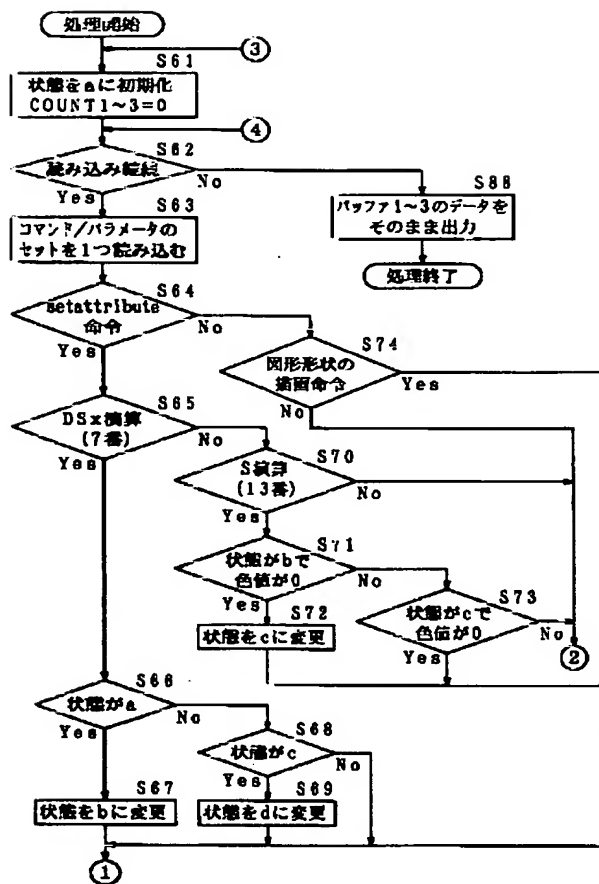
クリップ領域を指定  
(楕円形状を指定)

グラデーション図形  
クリップ領域に上書き

【図14】



【図13】



フロントページの続き

(72)発明者 小林 邦彦  
 神奈川県海老名市本郷2274番地 富士ゼロ  
 ックス株式会社内  
 (72)発明者 関根 弘  
 神奈川県海老名市本郷2274番地 富士ゼロ  
 ックス株式会社内

F ターム(参考) 5B057 CA18 CB01 CB16 CB18 CE18  
 CH01 CH11 CH18 DA08 DA17  
 DB06 DC25  
 5C076 AA02 AA26 AA40 BA05 BA06  
 BA10 CA12